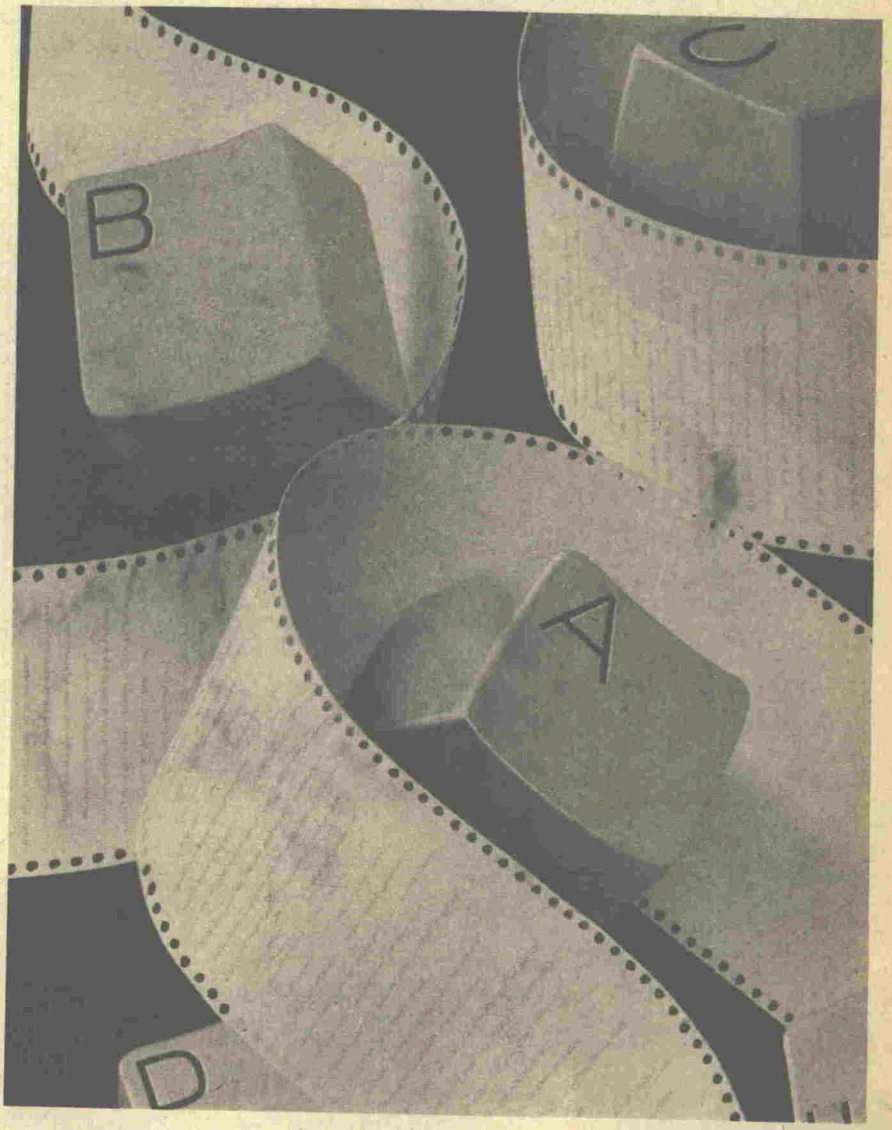
ขจร โรจนเมธินทร์

SOFT MAIL

โปรแกรม TIMEDATE
นาฬิกาตั้งเวลาสารพัดแบบ

สำหรับสอบถามหรือแก้ไขวันที่และเวลาในเวิร์ดสตาร์ผ่านทางคีย์บอร์ดแมคโครให้แสดงผลตามที่ต้องการ



คงเนื่องจากการใช้เวิร์ดสตาร์ที่อำนวยความสะดวกเรื่องการพิมพ์วันที่และเวลา โดยกด Esc แล้วตามด้วย @ หรือ ! เวิร์ดสตาร์ก็จะแทรกวันที่หรือเวลาให้ในตำแหน่งที่เคอร์เซอร์อยู่ ผู้เขียนเลยคิดว่าน่าจะแทรกวันที่ภาษาไทยได้บ้าง โปรแกรม TIMEDATE ที่เสนอในบทความนี้จะเป็นคีย์บอร์ดแมคโครซึ่งจะแทรกวันที่หรือเวลาทั้งภาษาไทยและอังกฤษในแบบต่าง ๆ ลง ณ ตำแหน่งของเคอร์เซอร์

การใช้งาน TIMEDATE

หลังจากที่โหลด TIMEDATE ที่ดอสพรอมต์แล้ว TIMEDATE จะซ่อนตัวอยู่บนหน่วยความจำ และคอยตรวจสอบการกดคีย์บอร์ดว่าเป็นไปตามตารางที่ 1 หรือไม่ ถ้าใช่โปรแกรมก็จะทำงานตามความหมายของคีย์นั้น

การตั้งวันที่ หลังจากที่โหลด TIMEDATE แล้วจะต้องรัน TIMEDATE อีกครั้งหนึ่งเพื่อแก้ไขวันที่ซึ่งตั้งใหม่ด้วยเสมอ เนื่องจาก TIMEDATE จะอ่านวันที่มาเก็บไว้ในตัวโปรแกรมขณะรันครั้งแรกเท่านั้น

การกดคีย์	ความหมายของแมคโคร	ตัวอย่าง
Ctrl+LeftShift+F1	แสดงเวลาแบบ 12 ชั่วโมง	9 : 30 P.M.
Ctrl+LeftShift+F2	แสดงเวลาแบบ 24 ชั่วโมง	21 : 30
Ctrl+LeftShift+F3	แสดงวันที่แบบยาว	January 2, 1990
Ctrl+LeftShift+F4	แสดงวันที่แบบสั้น	1-2-90
Ctrl+LeftShift+F5	แสดงวันที่ภาษาไทย	2 มกราคม พ.ศ. 2533

ตารางที่ 1 แสดงความหมายและตัวอย่างของคีย์บอร์ดแมคโคร

TIMEDATE ทำงานอย่างไร

เมื่อเริ่มรัน TIMEDATE โปรแกรมจะเรียกอินเตอร์รัพต์ 1Ah ฟังก์ชัน 1928h เพื่อตรวจสอบว่า TIMEDATE ถูกโหลดหรือยัง ถ้าโหลดแล้วค่าใน AX ที่คืนกลับมาจะเป็น 2819h และค่าใน ES จะเป็นค่าโค้ดเซกเมนต์ของ TIMEDATE ที่อยู่บนหน่วยความจำ ซึ่งจะนำมาใช้ในการแก้ไขวันที่หลังจากมีการตั้งใหม่

จากนั้นโปรแกรมจึงเรียกฟังก์ชัน 2Ah ของดอสเพื่อนำวันที่มาเปลี่ยนให้เป็นสตริง 3 แบบ คือ วันที่แบบยาว วันที่แบบสั้น วันที่ภาษาไทย เก็บไว้ที่ LONG_DATE, SHORT_DATE, THAI_DATE ตามลำดับ และหากเป็นการโหลด TIMEDATE ครั้งแรก โปรแกรมจะแก้ไขแอดเดรสของอินเตอร์รัพต์ 9h, 1Ah, 1Ch ให้ชี้มาที่ KB_INT, CLOCK_INT, TIMER_INT โดยที่โพรซีเดอร์ KB_INT จะทำหน้าที่คอยตรวจสอบการกดคีย์ว่าเป็นคีย์ที่ปลุกการทำงานของ TIMEDATE หรือไม่ ซึ่งหากมีการกดคีย์ดังกล่าว ตัวแปร BUFFER_PTR จะถูกชี้ไปที่ LONG_DATE, SHORT_DATE หรือ THAI_DATE ถ้าเป็นการสอบถามวันที่ แต่หากเป็นการสอบถามเวลาโปรแกรมจะเรียกอินเตอร์รัพต์ 1Ah ฟังก์ชันศูนย์เพื่อนำค่า Clock count มาคำนวณเป็นเวลา จากนั้นก็จะสร้างสตริงของเวลาแบบ 24 ชั่วโมง หรือ 12 ชั่วโมง ทั้งนี้ขึ้นอยู่กับคีย์ที่เลือกไว้ แล้วจะกำหนดค่าให้ตัวแปร BUFFER_PTR ชี้มาที่ TIME_BUFFER

หน้าที่สุดท้ายของ TIMEDATE ก็คือ การนำค่าของสตริงที่ BUFFER_PTR ชี้ไปใส่ในคีย์บอร์ดบัฟเฟอร์ที่อยู่ภายในเนื้อที่ของไบออส ซึ่งทำหน้าที่โดยโพรซีเดอร์ FILL_BUFFER ที่สำคัญหากสตริงของวันที่ยาวกว่าขนาดของคีย์บอร์ดบัฟเฟอร์ (16 เวิร์ด) จะทำให้คีย์บอร์ดบัฟเฟอร์เต็มก่อนที่จะใส่ค่าของสตริงได้ครบถ้วน ดังนั้นจึงจำเป็นต้องหยุดการทำงานของ TIMEDATE เพื่อเปิดโอกาสให้โปรแกรมอื่นอ่านค่าจากคีย์บอร์ดบัฟเฟอร์เสียก่อน จากนั้นค่อยย้อนมาใส่ค่าสตริงในคีย์บอร์ดบัฟเฟอร์อีกครั้งหนึ่ง ถ้าต้องการกลับไปทำงานใน FILL_BUFFER ที่ทำค้างไว้นั้นเราทำได้โดยใช้อินเตอร์รัพต์ 1Ch ซึ่งขณะนี้ชี้มาที่ TIMER_INT ให้ตรวจสอบค่าของ BUFFER_FLAG หากพบว่ามีค่าเป็นหนึ่งแสดงว่า FILL_BUFFER ยังทำงานไม่เสร็จ TIMER_INT ก็จะเรียกให้ FILL_BUFFER มาทำงานต่อ และเมื่อใส่ค่าของสตริงในคีย์บอร์ดบัฟเฟอร์เรียบร้อย ค่าของ BUFFER_FLAG จะเป็นศูนย์

คำนวณเวลาจากค่า Clock count

เครื่องไอบีเอ็มพีซีจะควบคุมเวลาของระบบด้วยชิพ 8253 โดยจะเพิ่มค่า Clock count ที่มีขนาด 3 เวิร์ดซึ่งเก็บอยู่ที่แอดเดรส TIMER_LOW (40:6C) และ TIMER_HIGH (40:6E) ภายในไบออส โดยใน 1 วินาทีจะเพิ่มค่าประมาณ 18.2 ครั้ง หรือประมาณ 65,543 ครั้งในหนึ่งชั่วโมง สังเกตได้ว่าใกล้เคียงกับ 65,536 (10000h) ดังนั้นจึงพอจะคาดคะเนได้ว่าค่าของ TIMER_HIGH (เวิร์ดสูง) จะถูกเพิ่มค่าขึ้นทุกหนึ่งชั่วโมง นำค่าที่คืนจากอินเตอร์รัพต์ 1Ah ใน CX มาหารด้วย 24 ก็จะได้ค่าชั่วโมงของเวลา

วิธีการหาค่านาที เนื่องจากค่า Clock count ถูกเพิ่มค่าขึ้น 65,536 ครั้งในหนึ่งชั่วโมง เพราะฉะนั้นถ้านำค่า TIMER_LOW (เวิร์ดต่ำ) มาหารด้วย 65,536 แล้วคูณด้วย 60 ก็จะได้ค่านาที สำหรับในการคำนวณจริง ๆ ให้นำค่า TIMER_LOW ที่อยู่ใน DX มาคูณกับ 60 ผลลัพธ์ของการคูณจะเป็นดับเบิ้ลเวิร์ดใน AX:DX ซึ่งการหารด้วย 65,536 คือการเลื่อนบิตของตัวตั้งไป 16 ครั้ง ดังนั้นเวิร์ดต่ำใน AX จะถูกแทนที่ด้วยเวิร์ดสูงของ DX เราจึงสามารถใช้ค่าใน DL ที่ได้หลังจากการคูณ โดยไม่ต้องหารเป็นค่านาทีของเวลา

ก่อนจบ

TIMEDATE จัดเป็นคีย์บอร์ดแมคโครทำหน้าที่แทรกวันที่หรือเวลาแบบต่าง ๆ ที่ตำแหน่งเคอร์เซอร์เมื่อมีการกดคีย์ตามตารางที่ 1 กรณีที่มีการตั้งวันที่ใหม่หลังจากโหลด TIMEDATE แล้ว ผู้ใช้จะต้องรัน TIMEDATE อีกครั้งเพื่อแก้ไขวันที่ให้ถูกต้อง หากลองกดคีย์ Ctrl+LeftShift+F5 แต่อ่านวันที่ภาษาไทยไม่ได้แสดงว่าเครื่องคอมพิวเตอร์ใช้รหัสภาษาไทยไม่ตรงกับโปรแกรม สำหรับโปรแกรมของผู้เขียนใช้รหัสของมหาวิทยาลัยเกษตรศาสตร์ ดังนั้นผู้ใช้เพียงแค่แก้ไขโปรแกรมบริเวณ MONTH_THAI เสียใหม่ให้ตรงตามรหัสที่ใช้บนเครื่องก็จะสามารถใช้ TIMEDATE ได้ดังใจนึก

โปรแกรม TIMEDATE
สอบถามวันที่และเวลาด้วยคีย์บอร์ด

```
 1 left_shift     equ 02h                 ;shift code for Left shift key
 2 ctrl_key       equ 04h                 ;shift code for Ctrl shift key
 3 f1_key         equ 3Bh                 ;scan code for F1 key
 4 f2_key         equ 3Ch                 ; ,,        ,, F2  ,,
 5 f3_key         equ 3Dh                 ; ,,        ,, F3  ,,
 6 f4_key         equ 3Eh                 ; ,,        ,, F4  ,,
 7 f5_key         equ 3Fh                 ; ,,        ,, F5  ,,
 8 eoi            equ 20h                 ;EOI value
 9 int_port       equ 20h                 ;8259 port address
10 kb_data        equ 60h                 ;keyboard data port
11 kb_ctrl        equ 61h                 ;keyboard control port
12 ;
13 bios_data      segment at 40h          ;BIOS data area
14                org 1Ah
15 buffer_head    dw  ?                   ;head of buffer
16 buffer_tail    dw  ?                   ;tail of buffer
17                org 80h
18 buffer_start   dw  ?                   ;start keyboard buffer
19 buffer_end     dw  ?                   ;end
20 bios_data      ends
21 ;
22 code           segment para public 'code'
23                assume cs:code
24                org 100h
25 begin:         jmp init1
26 ;
27                db  0Dh,"TimeDate Utility File !",1Ah
28 ;
29 long_date      db  19 dup (?)          ;long date string
30 short_date     db   9 dup (?)          ;short date string
31 thai_date      db  24 dup (?)          ;thai date string
32 time_buffer    db  11 dup (?)          ;time buffer
33 hour           db  ?                   ;current hour
34 minute         db  ?                   ;current minute
35 buffer_flag    db  0                   ;status of output routine
36 buffer_ptr     dw  ?                   ;index to output buffer
37 time_text      db  20h,'?.M.',0        ;AM/PM
38 old_int_9      label dword             ;old int 9
39 old_kbd_int    dw  2 dup (?)
40 old_int_1Ch    label dword             ;old int 1Ch
41 old_timer_int  dw  2 dup (?)
42 old_int_1Ah    label dword             ;old int 1Ah
43 old_clock_int  dw  2 dup (?)
44 ;
45 ;-------------------------------------------------------------------
46 ; TIMER_INT will be executed instead of the old one.
47 ;-------------------------------------------------------------------
```

```
48 timer_int     proc near
49               pushf
50               call old_int_1Ch          ;call the old int 1Ch
51               cmp  buffer_flag,0        ;do we waiting output?
52               je   timer_exit           ;no
53               push ax
54               push bx
55               push dx
56               push si
57               call fill_buffer          ;output to keyboard buffer
58               pop  si
59               pop  dx
60               pop  bx
61               pop  ax
62 timer_exit:   iret
63 timer_int     endp
64 ;
65 ;---------------------------------------------------------------------------
66 ; CLOCK_INT will be executed in order to check a installation.
67 ;---------------------------------------------------------------------------
68 clock_int     proc near
69               cmp  ax,1928h             ;new function of int 1Ah?
70               je   clock1               ;yes, then do my process
71               jmp  old_int_1Ah          ;no, then go to the old handler
72 clock1:        xchg ah,al                ;swap AH and AL
73               push cs                   ;return CS in ES
74               pop  es
75               iret                      ;exit
76 clock_int     endp
77 ;
78 ;---------------------------------------------------------------------------
79 ; KB_INT will be executed every hitting of any key.
80 ;---------------------------------------------------------------------------
81 kb_int        proc near
82               sti                       ;enable interrupt
83               push ax
84               mov  ah,2                 ;get status of shift key
85               int  16h
86               test al,ctrl_key          ;is the ctrl shift key pressed?
87               jz   kb_exit              ;no, go to the old handler
88               test al,left_shift        ;is the left shift key pressed?
89               jz   kb_exit              ;no
90               in   al,kb_data           ;get scan code
91               cmp  al,f1_key            ;is F1?
92               je   kb_int1              ;yes
93               cmp  al,f2_key            ;is F2?
94               je   kb_int1              ;yes
95               cmp  al,f3_key            ;is F3?
96               je   kb_int1              ;yes
97               cmp  al,f4_key            ;is F4?
98               je   kb_int1
99               cmp  al,f5_key            ;is F5?
100              je   kb_int1
101 kb_exit:     pop  ax
102              jmp  old_int_9
103 ;
104 kb_int1:     push bx
105              push cx
106              push dx
107              push si
108              push di
109              push ds
```

```
110                push es
111                push ax                     ;save scan code
112                in   al,kb_ctrl             ;get keyboard control byte
113                mov  ah,al                  ;save in AH
114                or   al,80h                 ;set high bit
115                out  kb_ctrl,al             ;send to control port
116                mov  al,ah                  ;get original byte
117                out  kb_ctrl,al             ;out to enable byte
118                pop  ax
119                push cs
120                pop  es                     ;ES point to Code
121                push cs
122                pop  ds                     ;DS point to Code
123                assume ds:code
124                cmp  al,f2_key              ;is Ctrl+LeftShift+F2 pressed?
125                jle  kb_int2
126 ;
127                mov  buffer_ptr,offset long_date   ;point to long date
128                cmp  al,f3_key                     ;is Ctrl+LeftShift+F3 pressed?.
129                je   kb_int6                       ;yes
130                mov  buffer_ptr,offset short_date  ;point to short date
131                cmp  al,f4_key                     ;is Ctrl+LeftShift+F4 pressed?
132                je   kb_int6                       ;yes
133                mov  buffer_ptr,offset thai_date   ;point to thai date
134                jmp  kb_int6
135 ;
136 kb_int2:       mov  bh,al                  ;save scan code
137                mov  ah,0                   ;get clock count from BIOS
138                int  1Ah                    ;time count in DX:CX
139                mov  ax,cx                  ;get hour part
140                mov  bl,24
141                div  bl                     ;divide by 24 hrs
142                mov  hour,ah                ;remainder is hour
143                mov  ax,60                  ;multiply DX by 60
144                mul  dx
145                mov  minute,dl              ;save minute
146                cld
147                lea  di,time_buffer         ;set buffer pointer
148                mov  al,hour                ;get hour
149                cmp  bh,f2_key              ;24 hours form?
150                je   kb_int4                ;yes
151                cmp  al,0                   ;is hour zero?
152                jne  kb_int3                ;no
153                add  al,12                  ;change to 12
154 kb_int3:       cmp  al,12                  ;hour > 12?
155                jbe  kb_int4                ;no
156                sub  al,12                  ;yes, subtract 12
157 kb_int4:       mov  bl,30h
158                call bin2ascii              ;convert to text
159                mov  al,':'
160                stosb
161                mov  al,minute              ;get minute
162                mov  bl,0                   ;print zero before
163                call bin2ascii
164                cmp  bh,f2_key              ;24 hours form?
165                jne  kb_int4x               ;yes
166                xor  al,al                  ;set string terminator
167                stosb
168                jmp  short kb_int5x         ;go to fill kb_buffer
169 kb_int4x:      mov  time_text[1],'P'       ;set to P.M.
170                cmp  hour,11                ;hour > 11?
171                ja   kb_int5                ;yes
```

```
172                 mov  time_text[1],'A'      ;set to 'A.M.'
173 kb_int5:        lea  si,time_text          ;add to time string
174                 mov  cx,6                  ;six char.
175                 rep  movsb
176 kb_int5x:       mov  buffer_ptr,offset time_buffer ;set for output
177 ;
178 kb_int6:        call fill_buffer           ;output done
179                 cli                        ;disable interrupt
180                 mov  al,eoi                ;end interrupt
181                 out  int_port,al
182                 pop  es
183                 pop  ds
184                 pop  di
185                 pop  si
186                 pop  dx
187                 pop  cx
188                 pop  bx
189                 pop  ax
190                 iret
191 kb_int         endp
192 ;
193 ;-------------------------------------------------------------------------
194 ; BIN2ASCII converts a binary digit less than 100 into its text form.
195 ; Entry: ES:DI - buffer address
196 ;        AL    - byte value
197 ;        BL    - 0 (print zero before), 30h (not print zero)
198 ;-------------------------------------------------------------------------
199 bin2ascii       proc near
200                 aam                        ;convert AL to BCD value in AX
201                 add  ax,3030h              ;binary to ASCII
202                 cmp  ah,bl                 ;not print zero?
203                 je   bin1                  ;yes
204                 xchg ah,al                 ;get first digit in AL
205                 stosb                      ;print first
206                 xchg ah,al                 ;restore
207 bin1:           stosb                      ;print second
208                 ret
209 bin2ascii      endp
210 ;
211 ;-------------------------------------------------------------------------
212 ; FILL_BUFFER outputs a string of bytes to the keyboard buffer.
213 ; Entry: BUFFER_PTR - offset address of next byte to transmit
214 ;-------------------------------------------------------------------------
215 fill_buffer    proc near
216                 push ds
217                 mov  ax,bios_data          ;point to bios data segment
218                 mov  ds,ax
219                 assume ds:bios_data
220                 mov  bx,buffer_tail        ;get position of buffer tail
221                 mov  si,buffer_ptr         ;get value of buffer index
222                 xor  ah,ah                 ; AH = 0
223 fill1:          mov  al,cs:[si]            ;get a char.
224                 or   al,al                 ;end of string?
225                 je   done                  ;yes
226                 mov  dx,bx                 ;get buffer tail into DX
227                 add  dx,2                  ;next position
228                 cmp  dx,buffer_end         ;do we need to wrap around?
229                 jne  fill2                 ;no
230                 mov  dx,buffer_start       ;wrap around to start
231 fill2:          cmp  dx,buffer_head        ;head = tail?
232                 je   buffer_full           ;yes, buffer is full
233                 mov  [bx],ax               ;deposit character into buffer
```

```
234                mov  bx,dx                ;change keyboard buffer pointer
235                mov  buffer_tail,bx       ;update bios buffer tail
236                inc  si                   ;next
237                inc  buffer_ptr
238                jmp  short fill1
239 buffer_full:   mov  buffer_flag,1        ;set flag to indicate not ok
240                jmp  short done1
241 done:          mov  buffer_flag,0        ;ok
242 done1:         pop  ds
243                assume ds:code
244                ret
245 fill_buffer    endp
246 ;
247 ;-------------------------------------------------------------------------
248 ; INITIALIZE will set the data area for the date and set address int 9, 1Ch
249 ;-------------------------------------------------------------------------
250 initialize     proc near
251 ;
252 month_text     db 7,'January  '          ;English text of month
253                db 8,'February '
254                db 5,'March    '
255                db 5,'April    '
256                db 3,'May      '
257                db 4,'June     '
258                db 4,'July     '
259                db 6,'August   '
260                db 9,'September'
261                db 7,'October  '
262                db 8,'November '
263                db 8,'December '
264 ;
265 month_thai     db 6,'มกราคม    '         ;Thai text of month
266                db 10,'กุมภาพันธ์'
267                db 6,'มีนาคม    '
268                db 6,'เมษายน    '
269                db 7,'พฤษภาคม   '
270                db 8,'มิถุนายน  '
271                db 7,'กรกฎาคม   '
272                db 7,'สิงหาคม   '
273                db 7,'กันยายน   '
274                db 6,'ตุลาคม    '
275                db 9,'พฤศจิกายน'
276                db 7,'ธันวาคม   '
277 ;
278 already_load  db 0                      ;flag for checking already install
279 day           db ?                      ;store day
280 month         db ?                      ;store month
281 year          db ?                      ;store English year
282 thai_yr       db ?                      ;store Thai year
283 text1         db ', 19'                 ;mask of English year
284 text2         db ' พ.ศ. 25'             ;mask of Thai year
285 ;
286 msg1          db "TimeDate < complete installation >, Copyright (C) 1990",0Dh,0Ah,'$'
287 msg2          db "TimeDate < previous installation >",0Dh,0Ah
288 msg3          db 0Ah,"Command Key : Ctrl + Left shift + F1 = 12 hours Time",0Dh,0Ah
289                db "              Ctrl + Left shift + F2 = 24 hours Time",0Dh,0Ah
290                db "              Ctrl + Left shift + F3 = Long Date",0Dh,0Ah
291                db "              Ctrl + Left shift + F4 = Short Date",0Dh,0Ah
292                db "              Ctrl + Left shift + F5 = Thai Date",0Dh,0Ah,'$'
293 ;
294 init1:        mov  ax,1928h             ;new function of int 1Ah
295                int  1Ah
```

```
296                cmp  ax,2819h              ;timekey already install if 2819h
297                jne  init2                 ;no
298                lea  dx,msg2               ;already message
299                mov  ah,9
300                int  21h
301                inc  already_load          ;set already load flag
302 ;
303 ;If TimeDate is already installed, ES points to the code segment of TimeDate.
304 ;
305 init2:         mov  ah,2Ah                ;get system date
306                int  21h
307                mov  ax,cx                 ;save year
308                sub  cx,1900               ;subtract 1900 from year
309                add  ax,543                ;convert to thai
310                sub  ax,2500
311                mov  thai_yr,al            ;store all data
312                mov  month,dh
313                mov  day,dl
314                mov  year,cl
315                cld
316 ;
317                dec  dh                    ;find table offset of month
318                mov  al,10
319                mul  dh                    ;(month-1)*10
320                mov  si,ax                 ;and it by SI
321                add  si,offset month_text
322                mov  cl,[si]               ;get length
323                inc  si
324                xor  ch,ch
325                lea  di,long_date          ;point ES:DI to long date
326                rep  movsb                 ;transfer to buffer
327                mov  al,20h                ;and space
328                stosb
329                mov  al,dl                 ;get day
330                mov  bl,30h                ;not print zero
331                call bin2ascii             ;add day to buffer
332                lea  si,text1              ;add ' 19'
333                mov  cx,4
334                rep  movsb
335                mov  al,year
336                mov  bl,0
337                call bin2ascii             ;add year to buffer
338                mov  al,0
339                stosb                      ;end of string
340 ;
341                lea  di,short_date         ;point ES:DI to short date
342                mov  al,month              ;get month
343                mov  bl,30h
344                call bin2ascii             ;to buffer
345                mov  al,'-'
346                stosb
347                mov  al,day                ;get day
348                mov  bl,30h
349                call bin2ascii             ;to buffer
350                mov  al,'-'
351                stosb
352                mov  al,year               ;get year
353                mov  bl,0
354                call bin2ascii             ;to buffer
355                mov  al,0                  ;end of string
356                stosb
357 ;
```

```
358                lea  di,thai_date          ;point ES:DI to thai date
359                mov  al,day                ;get day
360                mov  bl,30h                ;with not print zero
361                call bin2ascii             ;to buffer
362                mov  al,20h                ;add ' '
363                stosb
364                mov  dh,month              ;get month
365                dec  dh                    ;find table offset of month
366                mov  al,11
367                mul  dh                    ;(month-1)*11
368                mov  si,ax                 ;and it by SI
369                add  si,offset month_thai
370                mov  cl,[si]               ;get length
371                inc  si
372                xor  ch,ch
373                rep  movsb                 ;transfer to buffer
374                lea  si,text2
375                mov  cx,8                  ;length
376                rep  movsb
377                mov  al,thai_yr            ;get year
378                mov  bl,0
379                call bin2ascii             ;to buffer
380                mov  al,0                  ;end of string
381                stosb
382
383                cmp  already_load,0        ;is already installed?
384                jz   init3                 ;no, then do installation
385                int  20h                   ;yes, then terminate
386 ;
387 init3:        mov  ax,351Ch              ;get int 1Ch
388                int  21h
389                mov  old_timer_int,bx      ;save it
390                mov  old_timer_int[2],es
391                mov  ax,251Ch              ;set int 1Ch
392                lea  dx,timer_int          ;point to the new routine
393                int  21h
394                mov  ax,3509h              ;get int 9h
395                int  21h
396                mov  old_kbd_int,bx        ;save it
397                mov  old_kbd_int[2],es
398                mov  ax,2509h              ;set int 9h
399                lea  dx,kb_int             ;point to the new routine
400                int  21h
401                mov  ax,351Ah              ;get int 1Ah
402                int  21h
403                mov  old_clock_int,bx      ;save it
404                mov  old_clock_int[2],es
405                mov  ax,251Ah              ;set int 1Ah
406                lea  dx,clock_int          ;point to the new routine
407                int  21h
408                lea  dx,msg1               ;complete message
409                mov  ah,9
410                int  21h
411                lea  dx,msg3               ;command message
412                int  21h
413                lea  dx,initialize         ;end of resident address
414                int  27h
415 initialize     endp
416 ;
417 code           ends
418                end  begin
419
```